

CLAIM AMENDMENTS

1 1. (original) A method of providing concurrent access to a resource object, the method
2 comprising the computer-implemented steps of:
3 creating and storing a lock data structure for a particular resource object, the lock data
4 structure comprising data indicative of values for a resource object
5 identification, a lock type, and a version number related to a number of
6 changes to the resource object since the lock data structure was generated;
7 receiving a request from a requesting process for a requested lock type for access to
8 the particular resource object; and
9 determining whether to grant the request based on the requested lock type and the
10 lock type in the lock data structure.

1 2. (currently amended) A method as recited in Claim 1, further comprising the step of:
2 if it is determined to grant the request, then
3 bringing the value of the lock type in the data structure into agreement with the lock
4 type in the request;
5 generating a lock object having data indicative of the values for the resource object
6 identification, the lock type and the version number from the lock data
7 structure, and
8 returning the lock object to the requesting process.

1 3. (original) A method as recited in Claim 1, further comprising the steps of:

BT
C1

2 receiving a lock to be released having data indicative of values for the resource object
3 identification and the lock type and the version number;
4 determining whether the data indicative of the value for the lock type in the lock to be
5 released indicates an exclusive lock, and
6 if it is determined the data indicates the exclusive lock is to be released, then changing
7 the value for the version number in the lock data structure based on the value
8 of the version number in the lock to be released.

1 4. (currently amended) A method as recited in Claim 2 1, wherein:

2 the lock data structure further comprises a reference number;
3 said step of ~~generating~~ creating a lock data structure further comprises setting the
4 reference number to a predetermined initial value; and
5 said method further comprises, if it is determined to grant the request, then replacing
6 the value of the reference number in the lock data structure with a sum of the
7 value of the reference number in the lock data structure and a predetermined
8 reference change value.

1 5. (original) A method as recited in Claim 4, further comprising the steps of:

2 receiving a lock to be released having data indicating the particular resource object;
3 determining whether the reference number substantially equals the predetermined
4 initial value of the reference number; and
5 if it is determined the reference number does not substantially equal the
6 predetermined initial value, then replacing the value of the reference number
7 in the lock data structure with a difference substantially equal to the value of

BT
C1

8 the reference number in the lock data structure minus the predetermined
9 reference change.

1 6. (original) A method as recited in Claim 5, further comprising, if it is determined the
2 reference substantially equals the predetermined initial value, then deleting the lock
3 data structure for the particular resource object.

1 7. (canceled)

1 8. (previously presented) A method of updating a resource object using optimistic locks,
2 the method comprising the computer-implemented steps of:

3 receiving from a client process a request to update a particular resource object;

4 sending to a lock manager process a request for a first lock for access to the particular

5 resource object, the request including data indicating an optimistic lock type;

6 receiving the first lock for access to the particular resource object, the first lock

7 including data indicating the resource object, the optimistic lock type and a

8 first value for a version number related to a number of changes to the resource

9 object since the lock manager generated a lock data structure corresponding to

10 the resource object; and

11 updating the resource object by

12 sending to a lock manager process a request for a second lock for access to the

13 particular resource object, the request including data indicating the

14 resource object identification and an exclusive lock type;

15 receiving the second lock for access to the particular resource object, the

16 second lock including data indicating the resource object

BT
C1

identification, the exclusive lock type and a second value for the
version number;

determining whether the second value for the version number substantially
equals the first value for the version number; and
if the second value substantially equals the first value, then
committing an updated resource object to the resource, and
replacing the second value in the reference number in the second lock
with a third value of the version number, the third value
computed by adding the second value and a predetermined
version change value.

9. (original) The method as recited in Claim 8, further comprising, if the second value
does not substantially equal the first value, then sending a message to the client process, the
message indicating that the resource object was not updated.

10. (previously presented) The method as recited in Claim 8, further comprising sending
to the lock manager process a first release message to release the first lock.

11. (previously presented) The method as recited in Claim 10, further comprising sending
to the lock manager process a second release message to release the second lock.

12. (previously presented) The method as recited in Claim 9, further comprising sending
to the lock manager process a release message to release the second lock, the release message
including data indicating the third value of the version number in the second lock and the

BT
C1
4 exclusive lock type, wherein the third value of the version number is used by the lock
5 manager to replace the second value of the version number in the lock data structure.

1 13. (original) A computer-readable medium carrying one or more sequences of
2 instructions for providing concurrent access to a resource object, which instructions, when
3 executed by one or more processors, cause the one or more processors to carry out the steps
4 of:

5 creating and storing a lock data structure for a particular resource object, the lock data
6 structure comprising data indicative of values for a resource object
7 identification, a lock type, and a version number related to a number of
8 changes to the resource object since the lock data structure was generated;
9 receiving a request from a requesting process for a requested lock type for access to
10 the particular resource object; and
11 determining whether to grant the request based on the requested lock type and the
12 lock type in the lock data structure.

1 14. (previously presented) A computer-readable medium carrying one or more sequences
2 of instructions for updating a resource object, which instructions, when executed by one or
3 more processors, cause the one or more processors to carry out the steps of:

4 receiving from a client process a request to update a particular resource object;
5 sending to a lock manager process a request for a first lock for access to the particular
6 resource object, the request including data indicating an optimistic lock type;
7 receiving the first lock for access to the particular resource object, the first lock
8 including data indicating the resource object, the optimistic lock type and a

BY
CI

9 first value for a version number related to a number of changes to the resource
10 object since the lock manager generated a lock data structure corresponding to
11 the resource object; and
12 updating the resource object by
13 sending to a lock manager process a request for a second lock for access to the
14 particular resource object, the request including data indicating the
15 resource object identification and an exclusive lock type;
16 receiving the second lock for access to the particular resource object, the
17 second lock including data indicating the resource object
18 identification, the exclusive lock type and a second value for the
19 version number;
20 determining whether the second value for the version number substantially
21 equals the first value for the version number; and
22 if the second value substantially equals the first value, then
23 committing an updated resource object to the resource, and
24 replacing the second value in the reference number in the second lock
25 with a third value of the version number, the third value
26 computed by adding the second value and a predetermined
27 version change value.

- 1 15. (original) An apparatus for providing concurrent access to a resource object,
2 comprising:
3 a processor;

4 one or more stored sequences of instructions which, when executed by the processor,
5 cause the processor to carry out the steps of:
6 creating and storing a lock data structure for a particular resource object, the
7 lock data structure comprising data indicative of values for a resource
8 object identification, a lock type, and a version number related to a
9 number of changes to the resource object since the lock data structure
10 was generated;
11 receiving a request from a requesting process for a requested lock type for
12 access to the particular resource object; and
13 determining whether to grant the request based on the requested lock type and
14 the lock type in the lock data structure.

1 16. (previously presented) An apparatus for updating a resource object, comprising:
2 a processor;
3 one or more stored sequences of instructions which, when executed by the processor,
4 cause the processor to carry out the steps of:
5 receiving from a client process a request to update a particular resource object;
6 sending to a lock manager process a request for a first lock for access to the
7 particular resource object, the request including data indicating an
8 optimistic lock type;
9 receiving the first lock for access to the particular resource object, the first
10 lock including data indicating the resource object, the optimistic lock
11 type and a first value for a version number related to a number of

BT
C1

changes to the resource object since the lock manager generated a lock
data structure corresponding to the resource object; and
updating the resource object by
sending to a lock manager process a request for a second lock for
access to the particular resource object, the request including
data indicating the resource object identification and an
exclusive lock type;
receiving the second lock for access to the particular resource object,
the second lock including data indicating the resource object
identification, the exclusive lock type and a second value for
the version number;
determining whether the second value for the version number
substantially equals the first value for the version number; and
if the second value substantially equals the first value, then
committing an updated resource object to the resource, and
replacing the second value in the reference number in the
second lock with a third value of the version number,
the third value computed by adding the second value
and a predetermined version change value.

17. (original) An apparatus for providing concurrent access to a resource object,
comprising:
means for creating and storing a lock data structure for a particular resource object,
the lock data structure comprising data indicative of values for a resource

BT
C

5 object identification, a lock type, and a version number related to a number of
6 changes to the resource object since the lock data structure was generated;
7 means for receiving a request from a requesting process for a requested lock type for
8 access to the particular resource object; and
9 means for determining whether to grant the request based on the requested lock type
10 and the lock type in the lock data structure.

- 1 18. (previously presented) An apparatus for updating a resource object, comprising:
2 a means for receiving from a client process a request to update a particular resource
3 object;
4 a means for sending to a lock manager process a request for a first lock for access to
5 the particular resource object, the request including data indicating an
6 optimistic lock type;
7 a means for receiving the first lock for access to the particular resource object, the
8 first lock including data indicating the resource object, the optimistic lock type
9 and a first value for a version number related to a number of changes to the
10 resource object since the lock manager generated a lock data structure
11 corresponding to the resource object; and
12 a means for updating the resource object, including
13 a means for sending to a lock manager process a request for a second lock for
14 access to the particular resource object, the request including data
15 indicating the resource object identification and an exclusive lock type;
16 a means for receiving the second lock for access to the particular resource
17 object, the second lock including data indicating the resource object

BT
C1

18 identification, the exclusive lock type and a second value for the
19 version number;
20 a means for determining whether the second value for the version number
21 substantially equals the first value for the version number;
22 a means for committing an updated resource object to the resource if the
23 second value substantially equals the first value; and
24 a means for replacing the second value in the reference number in the second
25 lock with a third value of the version number if the second value
26 substantially equals the first value, the third value computed by adding
27 the second value and a predetermined version change value.

1 19. (previously presented) The computer-readable medium as recited in Claim 14,
2 wherein the instructions, when executed by one or more processors, cause the one or more
3 processors to carry out the step of:
4 if the second value does not substantially equal the first value, then sending a message
5 to the client process, the message indicating that the resource object was not
6 updated.

1 20. (previously presented) The computer-readable medium as recited in Claim 14,
2 wherein the instructions, when executed by one or more processors, cause the one or more
3 processors to carry out the step of:
4 sending to the lock manager process a first release message to release the first lock.

BT
C1

1 21. (previously presented) The computer-readable medium as recited in Claim 20,
2 wherein the instructions, when executed by one or more processors, cause the one or more
3 processors to carry out the step of:

4 sending to the lock manager process a second release message to release the second
5 lock.

1 22. (previously presented) The computer-readable medium as recited in Claim 19,
2 wherein the instructions, when executed by one or more processors, cause the one or more
3 processors to carry out the step of:

4 sending to the lock manager process a release message to release the second lock, the
5 release message including data indicating the third value of the version
6 number in the second lock and the exclusive lock type, wherein the third value
7 of the version number is used by the lock manager to replace the second value
8 of the version number in the lock data structure.

1 23. (previously presented) The apparatus as recited in Claim 16, wherein the instructions,
2 when executed by one or more processors, cause the one or more processors to carry out the
3 step of:

4 if the second value does not substantially equal the first value, then sending a message
5 to the client process, the message indicating that the resource object was not
6 updated.

24. (previously presented) The apparatus as recited in Claim 16, wherein the instructions, when executed by one or more processors, cause the one or more processors to carry out the step of:

sending to the lock manager process a first release message to release the first lock.

25. (previously presented) The apparatus as recited in Claim 24, wherein the instructions, when executed by one or more processors, cause the one or more processors to carry out the step of:

sending to the lock manager process a second release message to release the second lock.

26. (previously presented) The apparatus as recited in Claim 23, wherein the instructions, when executed by one or more processors, cause the one or more processors to carry out the step of:

sending to the lock manager process a release message to release the second lock, the release message including data indicating the third value of the version number in the second lock and the exclusive lock type, wherein the third value of the version number is used by the lock manager to replace the second value of the version number in the lock data structure.

27. (previously presented) The apparatus as recited in Claim 18, further comprising: means for sending a message to the client process if the second value does not substantially equal the first value, the message indicating that the resource object was not updated.

BT
C

1 28. (previously presented) The apparatus as recited in Claim 18, further comprising:
2 means for sending to the lock manager process a first release message to release the
3 first lock.

1 29. (previously presented) The apparatus as recited in Claim 28, further comprising:
2 means for sending to the lock manager process a second release message to release
3 the second lock.

1 30. (previously presented) The apparatus as recited in Claim 27, further comprising:
2 means for sending to the lock manager process a second release message to release the
3 second lock, the second release message including data indicating the third value
4 of the version number in the second lock and the exclusive lock type, wherein
5 the third value of the version number is used by the lock manager to replace the
6 second value of the version number in the lock data structure.